Faculty of Civil Engineering
Belgrade
Master Study
COMPUTATIONAL ENGINEERING

Faculty of Civil Engineering and Architecture
Niš
Doctoral Study

LECTURES

LESSON IX

9. Numerical Differentiation and Integration

9.1 Numerical Differentiation

In this section the numerical differentiation of real functions defined on $[a, b]$ will be considered.

9.1.1. Introduction

The need for numerical differentiation appears in following cases:

a. When values of function are known only on discrete set of points on $[a, b]$, i.e. function f is given in tabular form;
b. When analytical expression for function f is rather complicated. Numerical differentiation is chiefly based on approximation of function f by function Φ on $[a, b]$, and then differentiating Φ desirable times. Thus, based on $f^{(k)}(x) \sim \Phi(x) \quad (a \le x \le b)$), we have

$$f^{(k)}(x) \sim \Phi(x) \quad (a \le x \le b;\ k = 1, 2, \ldots).$$

For function Φ are mostly taken algebraic interpolation polynomials, because being simple differentiating. Let Φ be interpolating polynomial of $n-$th degree, i.e.

$$\Phi(x) = P_n(x).$$

If known error $R_n(f; x)$ of approximation polynomial

$$f(x) = P_n(x) + R_n(f; x) \quad (a \le x \le b) \tag{9.1.1.1}$$

it is possible to estimate error in formula for differentiation, i.e. from (9.1.1.1) it follows

$$f^{(k)}(x) = P_n^{(k)}(x) + R_n^{(k)}(f; x) \quad (a \le x \le b)$$

It is meaningful to take for order of derivative only $k < n$.

It is obvious that numerical differentiation has smaller accuracy than interpolation. So, for example, for interpolation is error in nodes equal to zero, what is not in case of differentiation.

9.1.2. Formulas for numerical differentiation

If known values of function f on set of equidistant points $\{x_0, x_1, \ldots, x_m\} \in [a, b]$, with step h, let

$$f_k = f(x_k) = f(x_0 + k \cdot h) \qquad (k = 0, 1, \ldots, m).$$

Construct over set $\{x_i, x_{i+1}, \ldots, x_{i+n}\}$ $(0 \le i \le m-n)$ first Newton interpolation polynomial (see Chapter 7, Finite Difference Calculus)

(9.1.2.1)
$$P_n(x) = f_i + p\,\Delta f_i + \frac{p\,(p-1)}{2!}\Delta^2 f_i + \frac{p\,(p-1)(p-2)}{3!}\Delta^3 f_i + \ldots + \frac{p\,(p-1)\ldots(p-n+1)}{n!}\Delta^n f_i,$$

where $p = (x - x_i)/h$. Because $P_n'(x) = \dfrac{1}{h}\dfrac{dP_n(x)}{dp}$, by differentiation (9.1.2.1) we get

(9.1.2.2)
$$P_n'(x) = \frac{1}{h}\big(\Delta f_i + \frac{2p-1}{2}\Delta^2 f_i + \frac{3p^2-6p+2}{6}\Delta^3 f_i + \ldots\big).$$

By further differentiation of (9.1.2.2) we get, in turn $P_n^{''}, P_n^{'''}$, and so on. For example,

(9.1.2.3)
$$P_n^{''}(x) = \frac{1}{h^2}\big(\Delta^2 f_i + (p-1)\Delta^3 f_i + \ldots\big).$$

For $x = x_i$, i.e. $p = 0$, formulas (9.1.2.2) and (9.1.2.3) reduce to

$$\begin{aligned} P_n'(x) &= \frac{1}{h}\big(\Delta f_i - \frac{1}{2}\Delta^2 f_i + \frac{1}{3}\Delta^3 f_i - \ldots + \frac{(-1)^{n-1}}{n}\Delta^n f_i\big),\\ P_n^{''}(x) &= \frac{1}{h^2}\big(\Delta^2 f_i - \Delta^3 f_i + \frac{11}{12}\Delta^4 f_i - \ldots\big). \end{aligned}$$

Instead of first Newton interpolation polynomial one can use other interpolation polynomials. For example, by differentiation Stirling polynomial

$$P_n(x) = f_i + \binom{p}{1}\delta\mu f_i + \frac{p}{2}\binom{p}{1}\delta^2 f_i + \binom{p+1}{3}\delta^3\mu f_i + \frac{p}{4}\binom{p+1}{3}\delta^4 f_i + \ldots,$$

where $x = x_i + p\cdot h$, we get

$$\begin{aligned} P_n'(x) &= \frac{1}{h}\big(\delta\mu f_i + p\,\delta^2 f_i + \frac{3p^2-1}{6}\delta^3\mu f_i + \frac{2p^3-p}{12}\delta^4 f_i + \ldots\big),\\ P_n^{''}(x) &= \frac{1}{h^2}\big(\delta^2 f_i + p\,\delta^3\mu f_i + \frac{6p^2-1}{12}\delta^4 f_i + \ldots\big),\\ P_n^{'''}(x) &= \frac{1}{h^3}\big(\delta^3\mu f_i + p\,\delta^4 f_i + \ldots\big),\ \text{etc.} \end{aligned}$$

In similar way, as for Newton's formulas, for $x = x_i$, i.e. $p = 0$, previous formulas reduce to

$$\begin{aligned} P_n'(x_i) &= \frac{1}{h}\big(\delta\mu f_i - \frac{1}{6}\delta^3\mu f_i + \ldots\big),\\ P_n^{''}(x_i) &= \frac{1}{h^2}\big(\delta^2 f_i - \frac{1}{12}\delta^4 f_i + \ldots\big),\ \text{etc.} \end{aligned}$$

Formulas for derivatives of function in interpolation nodes can also be obtained by formal application of operational calculus. Using development of operator D, and starting from definition $e^{hD} = E = 1 + \Delta$, i.e. $D^k = \dfrac{1}{h^k}\{\log(1+\Delta)\}^k$ by means of Stirling numbers of first kind, one gets k-th derivative in form (see [1], pp. 127-129)

$$f^{(k)}(x_i) = D^k f_i = \frac{1}{h^k}(\Delta^k f_i - \frac{k}{2}\Delta^{k+1} f_i + \frac{k(3k+5)}{24}\Delta^{k+2} f_i - \ldots).$$

For $k = 1$ we have

(9.1.2.4)
$$f'(x_i) = \frac{1}{h}\big(\Delta f_i - \frac{1}{2}\Delta^2 f_i + \frac{1}{3}\Delta^3 f_i - \ldots + \frac{(-1)^{n-1}}{n}\Delta^n f_i\big) + R_n'(f; x_i),$$

and error

(9.1.2.5)
$$R_n'(f; x_i) = \frac{(-1)^n}{n+1}h^n f^{(n+1)}(\xi_n) \qquad (x_i < \xi_n < x_{n+i}),$$

if $f \in C^{n+1}[a,b]$. By using backward operator ∇, in similar way like (9.1.2.4), we get

$$f'(x_i) = \frac{1}{h}\big(\nabla f_i - \frac{1}{2}\nabla^2 f_i + \frac{1}{3}\nabla^3 f_i + \ldots + \frac{1}{n}\nabla^n f_i\big) + R_n'(f;x_i),$$

and residue

$$R_n'(f;x_i) = \frac{1}{n+1}h^n f^{(n+1)}(\eta_n) \qquad (x_{i-n} < \eta_n < x_i),$$

if $f \in C^{n+1}[a,b]$. The corresponding formulas for $n = 1,2,3$ are

$$\begin{aligned} f'(x_i) &= \frac{1}{h}(f_i - f_{i-1}) + \frac{h}{2}f^{''}(\eta_1),\\ f'(x_i) &= \frac{1}{2h}(3f_i - 4f_{i-1} + f_{i-2}) + \frac{h^2}{3}f^{'''}(\eta_2),\\ f'(x_i) &= \frac{1}{6h}(11f_i - 18f_{i-1} + 9f_{i-2} - 2f_{i-3}) + \frac{h^3}{4}f^{IV}(\eta_3). \end{aligned}$$

Previous formulas for first derivative in node x_i are obviously asymmetric and are usually applied on the interval $[a,b]$ boundaries. Typical application of these formulas is at approximation of differentiable boundary conditions in contour problems of differential equations.

For nodes inside of segment $[a,b]$ is better to use symmetric differentiation formulas which are obtained by using operator of central difference (see [1], pp. 130-133). The symmetric formulas are (note that $f^{(k)}(x_i) = D^k f_i$)

(9.1.2.6) $$D = \frac{1}{h}\big(\delta - \frac{1^2}{2^2 3!}\delta^3 + \frac{1^2\cdot 3^2}{2^4 5!}\delta^5 - \frac{1^2\cdot 3^2\cdot 5^2}{2^6 7!}\delta^7 + \ldots\big),$$

which enables obtaining of Df_i only if we have values of function in the middle of interpolation nodes, i.e. $Df_{i+1/2}$, because of $\delta^{2j+1}f_i = \delta^{2j}(f_{i+1/2} - f_{i-1/2})$. Formula which uses only the function values in interpolation nodes is

(9.1.2.7) $$D = \frac{\mu}{h}\big(\delta - \frac{1^2}{3!}\delta^3 + \frac{1^2\cdot 2^2}{5!}\delta^5 - \frac{1^2\cdot 2^2\cdot 3^2}{7!}\delta^7 + \ldots\big).$$

From last formula one can get a set of formulas, of which we will give two first:

(1^0) $$Df_i = \frac{1}{h}\mu\delta f_i + r_1(f) = \frac{1}{2h}(f_{i+1} - f_{i-1}) + r_1(f),$$

where

$$r_1(f) = -\frac{1}{6}h^2 f^{'''}(\xi_1) \quad (x_{i-1} < \xi_1 < x_{i+1});$$

(2^0) $$Df_i = \frac{1}{h}(\mu\delta f_i - \frac{1}{6}\mu\delta^3 f_i) + r_2(f) = \frac{1}{h}(-f_{i+2} + 8f_{i+1} - 8f_{i-1} + fi - 2) + r_2(f),$$

where

$$r_2(f) = \frac{1}{30}h^4 f^V(\xi_1) \quad (x_{i-2} < \xi_2 < x_{i+2}).$$

For obtaining higher derivatives are used formula (9.1.2.6) for derivatives of even order and formula (9.1.2.7) for derivatives of odd order, what gives

$$\begin{aligned} D^2 &= \frac{1}{h^2}(\delta^2 - \frac{1}{12}\delta^4 + \frac{1}{90}\delta^6 - \frac{1}{560}\delta^8 + \frac{1}{3150}\delta^{10} - \ldots),\\ D^3 &= \frac{\mu}{h^3}(\delta^3 - \frac{1}{4}\delta^5 + \frac{7}{120}\delta^7 - \ldots),\\ D^4 &= \frac{1}{h^4}(\delta^4 - \frac{1}{6}\delta^6 + \frac{7}{240}\delta^8 - \ldots),\\ D^5 &= \frac{\mu}{h^5}(\delta^5 - \frac{1}{3}\delta^7 + \ldots), \quad \text{etc.} \end{aligned}$$

The most used and simplest formula for approximation of second derivative is (taking only the first member in expression for D^2)

$$D^2 f_i = \frac{1}{h^2}\delta^2 f_i + r(f) = \frac{1}{h^2}(f_{i+1} - 2f_i + f_{i-1}) + r(f).$$

where, under condition $f \in C^4[a, b]$, the residue is

$$r(f) = -\frac{h^2}{12} f^{IV}(\xi).$$

Note that by using numerical integration the approximate formulas for differentiation of analytical functions can be obtained.

9.2. Numerical integration - Quadrature formulas

9.2.1. Introduction

Numerical integration of functions is dealing with approximative calculation of definite integrals on the basis of the sets of values of function to be integrated, by following some formula.

Formulas for calculation of single integrals are called **quadrature formulas**. In similar way, formulas for double integrals (and multi-dimensional integrals, too) are called cubature formulas.

In our considerations, we will deal mainly with quadrature formulas.

The need for numerical integration appears in many cases. Namely, Newton-Leibnitz formula (9.2.1.1)

$$\int_a^b f(x)dx = f(b) - f(a), \tag{9.2.1.1}$$

where F is primitive function of function f, cannot always be applied. Note some of these cases:

1. Function F cannot be represented by finite number of elementary functions (for example, when $f(x) = e^{-x^2}$).
2. Application of formula (9.2.1.1) leads often to very complicated expression, even at calculation of integral of rather simple functions, e.g.

$$\int_a^b \frac{dx}{1+x^3} = \log \sqrt[3]{|a+1|} - \frac{1}{6}\log(a^2 - a + 1) + \frac{1}{\sqrt{3}} arctg \frac{a\sqrt{3}}{2-a}.$$

3. Integration of functions with values known on discrete set of points (obtained, for example, by experiments), is not possible by applying formula (9.2.1.1).

Large number of quadrature formulas are of form

$$\int_a^b f(x)dx \cong \sum_{k=0}^{n} A_k f_k, \tag{9.2.1.2}$$

where $f_k = f(x_k)$ $(a \leq x_0 < \ldots < x_n \leq b)$. If $x_0 = a$ and $x_n = b$, formula (9.1.1.2) is of closed kind, and in other cases is of open kind.

For integration of differentiable functions are used also formulas which have, in addition to function values, values of its derivatives. The formulas for calculation of integrals of form

$$\int_a^b p(x)f(x)dx,$$

where $x \to p(x)$ is given weight function, are also of concern.

One simple way for construction of quadrature formulas is founded on application of interpolation. Formulas obtained in this way are called **interpolating** quadrature formulas.

Let the values of function f in given points
$x_0, x_1, \ldots, x_n (\in [a, b])$ be $f_0, f_1, \ldots, f_n$ respectively, i.e.

$$f_k \equiv f(x_k) \qquad (k = 0, 1, \ldots, n).$$

On the basis of these data, we can construct Lagrange interpolation polynomial

$$P_n(x) = \sum_{k=0}^{n} f_k \frac{\omega(x)}{(x - x_k)\omega'(x_k)},$$

where $\omega(x) = (x - x_0)(x - x_1) \cdots (x - x_n)$.

Then

$$\int_a^b p(x)f(x)dx = \int_a^b p(x)P_n(x)dx + R_{n+1}(f),$$

i.e.

$$\int_a^b p(x)f(x)dx = \sum_{k=0}^{n} A_k f_k + R_{n+1}(f), \tag{9.2.1.3}$$

where we put

$$A_k = \int_a^b \frac{p(x)\omega(x)}{(x - x_k)\omega'(x_k)} dx \qquad (k = 0, 1, \ldots, n).$$

In formula (9.2.1.3), $R_{n+1}(f)$ is called residue (rest, residuum) of quadrature formula and represents error done by replacing of integral by finite sum. Index $n + 1$ in residue denotes that integral is approximate calculated based on values of function to be integrated in $n + 1$ points.

Denote with π_n set of all polynomials of degree not greater than n.

Because $f(x) = x^k \quad (k = 0, 1, \ldots, n)$, $f(x) \equiv P_n(x)$, we have $R_{n+1}(x^k) \equiv 0 \quad (k = 0, 1, \ldots, n)$, wherefrom we conclude that formula (9.2.1.3) is exact for every $f \in \pi_n$, regardless of choice of interpolation nodes x_k $(k = 0, 1, \ldots, n)$ and in this case we say that (9.2.1.3) has algebraic degree of accuracy n.

9.2.2. Newton-Cotes formulas

In this section we will develop quadrature formulas od closed type in which the interpolation nodes $x_k = x_0 + kh$ $(k = 0, 1, \ldots, n)$ are taken equidistantly with a step $h = \frac{b - a}{n}$.

If we introduce substitution $x - x_0 = ph$, we have

$$\omega(x) = (x - x_0)(x - x_1) \ldots (x - x_n) = h^{n+1} p(p - 1) \ldots (p - n) \tag{9.2.2.1}$$

and

$$\begin{aligned} \omega'(x_k) &= (x_k - x_0)(x_k - x_1) \ldots (x_k - x_{k-1})(x_k - x_{k+1}) \ldots (x - x_n) \\ &= h^n (-1)^{n-k} k!1(n - k)! \end{aligned} \tag{9.2.2.2}$$

By introducing notation for generalized degree $x^{(s)} = x(x - 1) \ldots (x - s + 1)$, based on (9.2.2.1), (9.2.2.2) and results from previous section, we get

$$A_k = \int\limits_0^n \frac{(-1)^{n-k} p^{(n+1)} h}{(p - k)k!(n - k)!} dp \quad (k = 0, 1, \ldots, n),$$

i.e.

$$A_k = (b-a)H_k \quad (k = 0, 1, \ldots, n),$$

where we put

(9.2.2.3) $$H_k \equiv H_k(n) = \frac{(-1)^{n-k}}{n!n}\binom{n}{k}\int_0^n \frac{p^{(n+1)}}{p-k}dp \quad (k = 0, 1, \ldots, n).$$

Coefficients H_k are known in literature as Newton-Cotes coefficients, and corresponding formulas

(9.2.2.4) $$\int\limits_{x_0=a}^{x_n=b} f(x)dx = (b-a)\sum_{k=0}^{n} H_k f(a + k\frac{b-a}{n}) \quad (k \in N)$$

as Newton-Cotes formulas.

Further on we will give survey of Newton-Cotes formulas for $n \le 8$. Here we use denotations $h = \frac{b-a}{n}$, $f_k = f(x_k)$ $(k = 0, 1, \ldots, n)$.

1. $n = 1$ (Trapezoid rule)

$$\int\limits_{x_0}^{x_1} f(x)dx = \frac{h}{2}(f_0 + f_1) - \frac{h^3}{12}f''(\xi_1);$$

2. $n = 2$ (Simpson's rule)

$$\int\limits_{x_0}^{x_2} f(x)dx = \frac{h}{3}(f_0 + 4f_1 + f_2) - \frac{h^5}{90}f^{IV}(\xi_2);$$

3. $n = 3$ (Simpson's rule $\frac{3}{8}$)

$$\int\limits_{x_0}^{x_3} f(x)dx = \frac{3h}{8}(f_0 + 3f_1 + 3f_2 + f_3) - \frac{3h^5}{80}f^{IV}(\xi_3);$$

4. $n = 4$ (Boole's rule)

$$\int\limits_{x_0}^{x_4} f(x)dx = \frac{2h}{45}(7f_0 + 32f_1 + 12f_2 + 32f_3 + 7f_4) - \frac{8h^7}{945}f^{VI}(\xi_4);$$

5. $n = 5$

$$\int\limits_{x_0}^{x_5} f(x)dx = \frac{5h}{288}(19f_0 + 75f_1 + 50f_2 + 50f_3 + 75f_4 + 19f_5) - \frac{275h^7}{12096}f^{(6)}(\xi_5);$$

6. $n = 6$

$$\int\limits_{x_0}^{x_6} f(x)dx = \frac{h}{140}(41f_0 + 216f_1 + 27f_2 + 272f_3 + 27f_4 + 216f_5 + 41f_6) - \frac{9h^9}{1400}f^8(\xi_6);$$

7. $n = 7$

$$\int\limits_{x_0}^{x_7} f(x)dx = \frac{7h}{17280}(751f_0 + 3577f_1 + 1323f_2 + 2989f_3 + 2989f_4 + 1323f_5 + 3577f_6 + 751f_7) - \frac{8183h^9}{518400}f^{(8)}(\xi_7);$$

8. $n = 8$

$$\int_{x_0}^{x_8} f(x)dx = \frac{4h}{14175}(989f_0 + 5888f_1 - 928f_2 + 10496f_3 - 4540f_4 + 10496f_5 - 928f_6 + 5888f_7 + 989f_8) - \frac{2368h^{11}}{467775}f^{(10)}(\xi_8);$$

where $\xi_k \in (x_0, x_k)$ $(k = 1, \ldots, 8)$.

In general case, the residue $R_{n+1}(f)$ is of form

$$R_{n+1}(f) = C_n h^m f^{(m-1)}(\xi_n) \quad (x_0 < \xi_n < x_n),$$

where $m = 2[\frac{n}{2}] + 3$. Given equality has a meaning if function $f \in C^{m-1}[a, b]$.

9.2.3. Generalized quadrature formulas

In order to compute value of integrals more accurate it is necessary to divide segment $[a, b]$ to the set of subsegments, and then to apply to each of them some quadrature formula. In this way we get generalized or composite formulas. In this section we will consider generalized formulas obtained on the basis of trapezoid or Simpson's formula.

Divide the segment $[a, b]$ on set of subsegments $[x_{i-1}, x_i]$ so that $x_i = a + ih$ and $h = (b - a)/n$.

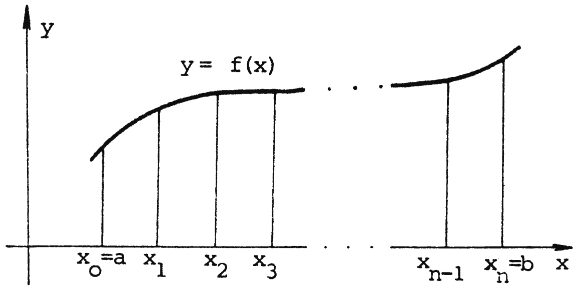

Figure 9.2.3.1

By applying the trapezoid formula on every subsegment, we get

$$\int_a^b f(x)dx = \sum_{i=1}^{n} \int_{x_{i-1}}^{x_i} f(x)dx = \sum_{i=1}^{n}(\frac{h}{2}(f_{i-1} + f_i) - \frac{h^3}{12}f''(\xi_i)),$$

i.e.

$$\int_a^b f(x)dx = T_n - \frac{h^3}{12}\sum_{i=1}^{n} f''(\xi_i),$$

where

$$T_n \equiv T_n(f; h) = h(\frac{1}{2}f_0 + f_1 + \cdots + f_{n-1} + \frac{1}{2}f_n)$$

and

$$x_{i-1} < \xi_i < x_i \quad (i = 1, 2, \ldots, n).$$

Theorem 9.2.3.1. *If $f \in C^2[a, b]$ the equality*

$$\int_a^b f(x)dx - T_n = -\frac{(b-a)^2}{12n^2}f''(\xi) \quad (a < \xi < b)$$

holds.

Quadrature formula

$$\int_a^b f(x)dx \cong T_n(f;h) \quad (h = \frac{b-a}{n})$$

is called generalized trapezoid formula.

Suppose now that $h = \dfrac{b-a}{2n}$, i.e. $x_i = a + ih \quad (i = 0, 1, \ldots, 2n)$ (See Fig. 9.2.3.2), and then apply Simpson's rule to subsegments

$$[x_0, x_2], \ldots, [x_{2n-2}, x_{2n}].$$

In this way we get generalized Simpson's formula

$$\int_a^b f(x)dx \cong S_n(f;h) \quad (h = \frac{b-a}{2n}),$$

where

$$S_n \equiv S_n(f,h) = \frac{h}{3}\{f_0 + 4(f_1 + \ldots + f_{2n-1}) + 2(f_2 + \ldots + f_{2n-2}) + f_{2n}\}.$$

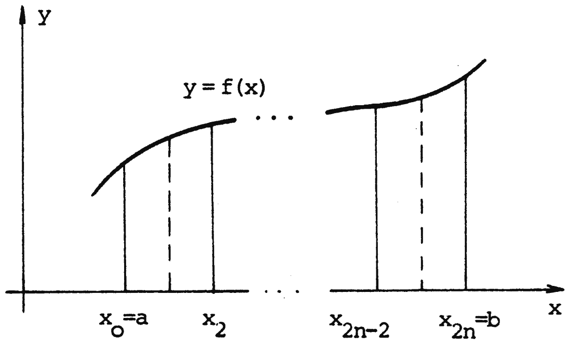

Figure 9.2.3.2

Theorem 9.2.3.2. *If $f \in C^4[a,b]$ the equality*

$$\int_a^b f(x)dx - S_n = -\frac{(b-a)^2}{2880n^4} f^{(IV)}(\xi) \quad (a < \xi < b)$$

holds.

9.2.4. Romberg integration

For calculation of definite integrals in practice is most frequently used generalized trapezoid formula in a special form, known as **Romberg integration**.

Denote with $T_k^{(0)}$ trapezoid approximation $T_n(f;h)$ $(n = 2^k)$, i.e. $h = \dfrac{(b-a)}{2^k}$). Romberg integration consists of construction of two-dimensional set $T_k^{(m)}$ $(m = 0, 1, \ldots, k;\ k = 0, 1, \ldots)$ using

$$T_k^{(m)} = \frac{4^m T_{k+1}^{(m-1)} - T_k^{(m-1)}}{4^m - 1}. \tag{9.2.4.1}$$

Using (9.2.4.1) one can construct so known T table

$$\begin{tikzcd}
T_0^{(0)} \arrow[r] & T_0^{(1)} \arrow[r] & T_0^{(2)} \arrow[r] & T_0^{(3)} \\
T_1^{(0)} \arrow[r] \arrow[ur] & T_1^{(1)} \arrow[r] \arrow[ur] & T_1^{(2)} \arrow[r] \arrow[ur] & \vdots \\
T_2^{(0)} \arrow[r] \arrow[ur] & T_2^{(1)} \arrow[r] \arrow[ur] & \vdots & \\
T_3^{(0)} \arrow[r] \arrow[ur] & \vdots & &
\end{tikzcd}$$

by taking $k = 0, 1, \ldots$ and $m = 1, 2, \ldots$. In first column of this table are in turn approximate values of integral I obtained by means of trapezoid formula with $h_k = (b - a)/2^k$ $(k = 0, 1, \ldots)$. Second column is obtained based on the first, using formula (9.2.4.1), third from second, and so on.

Iterative process, defined by (9.2.4.1) is the standard Romberg method for numerical integration. One can prove that series $\{T_k^{(m)}\}_{k\in N_0}$ and $\{T_k^{(m)}\}_{m\in N_0}$ (by columns and rows in T-table) converge to I. At practical application of Romberg integration, iterative process (9.2.4.1) is usually interrupted when $|T_0^{(m)} - T_0^{(m-1)}| \le \varepsilon$, where ε is in advance allowed error, and then as result is taken $I \cong T_0^{(m)}$.

9.2.5. Program realization

In this section we give program realization of Simpson's and Romberg integration.

Program 9.2.5.1.

For integration using generalized Simpson's formula the subroutine `INTEG` is written. Parameters in parameter list are of meaning explained in `C`- comments of subprogram source code. Function to be integrated is given in subroutine `FUN`, and depends on one parameter `Z`. By integer parameter `J` is provided simultaneous specifying more functions to integrate.

Subroutine `INTEG` is organized in this way that initial number of subsegments can be improved (by reduction of step h to $h/2$) up to `MAX=1000`. In case when relative difference in integrals values, obtained by steps h and $h/2$, is less than 10^{-5}, the calculation interrupts and value of integral calculated with the smallest step is taken as definitive value of integral. If this criterion cannot be fulfilled with less than `MAX` subsegments, the message `KBR=1` is printed (and in opposite case `KBR=0`).

As a test examples for this subroutine, the following integrals are taken:

$$\int_0^1 \frac{e^{z\,x}}{x^2+z^2}dx \qquad (z = 1.0(0.1)1.5),$$

$$\int_0^{1/2} \pi \sin(\pi z x)\, dx \qquad (z = 1.0(0.2)1.4)$$

$$\int_1^2 \frac{\log(x+z)}{z^2+e^x}\frac{\sin x}{x}dx \qquad (z = 0.0(0.1)0.5)$$

.

Subroutines, main program, and output listing are of form:

```
C==========================================================
C IZRACUNAVANJE ODREDENOG INTEGRALA FUNKCIJE F(X,Z,J)
C SIMPSONOVOM  FORMULOM
C==========================================================
      SUBROUTINE INTEG(A, B, S, F, J, KBR, Z)
C  A - LOWER LIMIT OF INTEGRAL
C  B - UPPER LIMIT OF INTEGRAL
C  S - VALUE OF INTEGRAL WITH ACCURACY  EPS=1.E-5
C  KBR - CONTROL NUMBER
C  KBR=0 INTEGRAL CORRECTLY COMPUTED
C  KBR=1 INTEGRAL NOT COMPUTED WITH SPECIFIED ACCURACY
C  Z - PARAMETAR OF INTEGRATED FUNCTION
C  INITIAL NUMBER OF SEGMENTS IS 2*MP MAXIMAL IS MAX=1000
       MP=15
       MAX=1000
       KBR=0
       N=2.*MP
       S0=0.
       SAB=F(A,Z,J)+F(B,Z,J)
       H=(B-A)/FLOAT(N)
       X=A
       S1=0.
       N2=N-2
       DO 5 I=2, N2, 2
       X=X+2.*H
5      S1=S1+F(X,Z,J)
10     S2=0.
       X=A-H
       N1=N-1
       DO 15 I=1, N1, 2
       X=X+2.*H
15     S2=S2+F(X,Z,J)
       S=H/3.*(SAB+2.*S1+4.*S2)
       REL=(S-S0)/S
       IF (ABS(REL)-1.E-5) 35,35,20
20     IF (N-MAX) 25,25,30
25     N=2*N
       H=0.5*H
C  BROJ PODEOKA SE UVECAVA DVA PUTA I
C IZRACUNAVA SE NOVA VREDNOST ZA S1
       S1=S1+S2
       S0=S
       GO TO 10
30     KBR=1
35     RETURN
       END
       FUNCTION FUN(X,Z,J)
       GO TO (10,20,30),J
10     FUN=EXP(Z*X)/(X*X+Z*Z)
       RETURN
20     PI=3.1415926535
       FUN=PI*SIN(PI*X*Z)
       RETURN
30     FUN=ALOG(X+Z)/(Z*Z+EXP(X))*SIN (X)/X
       RETURN
       END
       EXTERNAL FUN
       OPEN(8,File='Simpson.IN')
       OPEN(6,File='Simpson.out')
       WRITE(6,5)
5      FORMAT (1H1,2X, 'IZRACUNAVANJE VREDNOSTI INTEGRALA',
     1 ' PRIMENOM SIMPSONOVE FORMULE ' //14X,
     2 'TACNOST IZRACUNAVANJA EPS=1.E-5'
     3 ///11X,'J',4X,'DONJA',5X,'GORNJA',3X,'PARAMETAR',
     4 3X,' VREDNOST'/ 16X, 'GRANICA', 3X,'GRANICA',
     5  5X,'Z',7X,'INTEGRALA'//)
       DO 40 J=1,3
       READ (8,15) DG, GG, ZP, DZ, ZK
15     FORMAT(5F5.1)
       Z=ZP-DZ
18     Z=Z+DZ
       IF (Z.GT.ZK+0.000001) GO TO 40
       CALL INTEG (DG,GG,S,FUN,J,KBR,Z)
       IF(KBR) 20,25,20
```

```
20     WRITE (6,30)
30     FORMAT (/11X, 'INTEGRAL NIJE KOREKTNO IZRACUNAT'/)
       GO TO 18
25     WRITE (6,35) J,DG,GG,Z,S
35     FORMAT (11X,I1,F8.1,2F10.1,F15.6/)
       GO TO 18
40     CONTINUE
       STOP
       END
```

```
0.,1.,1.,0.1,1.5
0.,0.5,1.,0.2,1.4
1.,2.,0.,0.1,0.5
```

```
1  IZRACUNAVANJE VREDNOSTI INTEGRALA PRIMENOM SIMPSONOVE FORMULE
             TACNOST IZRACUNAVANJA EPS=1.E-5
          J    DONJA     GORNJA   PARAMETAR    VREDNOST
               GRANICA   GRANICA     Z       INTEGRALA
          1      .0        1.0       1.0       1.270724
          1      .0        1.0       1.1       1.153890
          1      .0        1.0       1.2       1.059770
          1      .0        1.0       1.3        .983069
          1      .0        1.0       1.4        .920013
          1      .0        1.0       1.5        .867848
          2      .0         .5       1.0       1.000000
          2      .0         .5       1.2       1.090848
          2      .0         .5       1.4       1.134133
          3     1.0        2.0        .0        .048047
          3     1.0        2.0        .1        .059595
          3     1.0        2.0        .2        .069940
          3     1.0        2.0        .3        .079052
          3     1.0        2.0        .4        .086920
          3     1.0        2.0        .5        .093558
```

Program 9.2.5.2.

Now we give program realization of Romberg integration in double arithmetic computer precision `DOUBLE PRECISION`. List in subroutine is of following meaning:

`DG` - lower limit of integral;

`GG` - upper limit of integral;

`FUN` - name of function subroutine which defines function to be integrated ;

`EPS` - demanded accuracy of computation;

`VINT` - value of integral for given accuracy `EPS`, if `KB=0`;

`KB` - control number (`KB=0` - integral correctly computed; `KB=1` - accuracy of computing not reached after `15` proposed steps, i.e. with numbers of subsegments 2^{15}). For testing of this subroutine is taken tabulating of function

$$F(x) = \int_0^x e^{-t^2} dt \qquad (x = 0.1(0.1)1.0),$$

with accuracy 10^{-5}. Routines codes and output listings are of form:

```
C==================================================================
C              ROMBERGOVA INTEGRACIJA
C==================================================================
      DOUBLE PRECISION GG, FUN, VINT
      EXTERNAL FUN
      open(6,file='romberg.out')
EPS=1.E-8
      WRITE (6,11)
11    FORMAT(1H0,5X,'X',7X,'INTEGRAL(0.,X)'/)
      DO 10 I=1, 10
      GG=0.1*I
      CALL ROMBI(0.D0,GG,FUN,EPS,VINT,KB)
      IF (KB) 5,15,5
    5 WRITE (6,20) GG
   20 FORMAT (5X,F3.1,4X,'TACNOST NE ZADOVOLJAVA'//)
      GO TO 10
   15 WRITE(6,25)GG,VINT
   25 FORMAT(5X,F3.1,4X,F14.9)
   10 CONTINUE
      STOP
      END
      SUBROUTINE ROMBI (DG,GG,FUN,EPS,VINT,KB)
      DOUBLE PRECISION FUN,VINT,T(15),DG,GG,H,A,POM,B,X
      KB=0
      H=GG-DG
      A=(FUN(DG)+FUN(GG))/2.
      POM=H*A
      DO 50 K=1, 15
      X=DG+H/2.
   10 A=A+FUN (X)
      X=X+H
      IF (X.LT.GG) GO TO 10
      T(K)=H/2.*A
      B=1.
      IF (K.EQ.1) GO TO 20
      K1=K-1
      DO 15 M=1, K1
      I=K-M
      B=4.*B
   15 T(I)=(B*T(I+1)-T(I))/(B-1.)
   20 B=4.*B
      VINT=(B*T(1)-POM)/(B-1.)
      IF(DABS(VINT-POM).LE.EPS) RETURN
      POM=VINT
   50 H=H/2.
      KB=1
      RETURN
      END
      FUNCTION FUN(X)
      DOUBLE PRECISION FUN,X
      FUN=DEXP(-X*X)
      RETURN
      END
```

```
   0  X       INTEGRAL(0.,X)
      .1        .099667666
      .2        .197365034
      .3        .291237887
      .4        .379652845
      .5        .461281012
      .6        .535153533
      .7        .600685674
      .8        .657669863
      .9        .706241521
     1.0        .746824138
```

9.2.6. On numerical computation of one class of double integrals

In this section we will point out to one way for approximate calculation of double integrals of form

$$\iint_G f(x,y)\,dxdy, \tag{9.2.6.1}$$

where area of integration is unit circle, i.e. $G = \{(x,y)\,|\,x^2+y^2 \le 1\}$. Namely, for numerical computation of the integral (9.2.6.1) in literature is known formula

$$\iint_G f(x,y)\,dxdy \cong \frac{\pi}{8}(2f(0) + \sum_{i=1}^{n} f(M_i)), \tag{9.2.6.2}$$

where O is origin, i.e. $0 = (0,0)$, and points M_i have polar coordinates

$$r_i = \sqrt{\frac{2}{3}}, \quad \Phi_i = \frac{\pi}{3}(i-1) \quad (i = 1, 2, \ldots, 6).$$

.

According to formula (9.2.6.2) we will realise program for computation of double integrals, with unit circle as area of integration. Program organization will be such that by function subroutine `EF` can be defined several different functions to be integrated f. Parameters in list of parameters are of following meaning:

`X` - value of argument x;

`Y` - value of argument y;

`K` - integer that defines different functions to be integrated.

Formula (9.2.6.2) is realized by subroutine `DVINT`, which parameters in list are of following meaning:

`EF` - name of function subroutine;

`K` - integer with same meaning like in subroutine `EF`;

`VRINT` - computed value of integral, obtained by using formula (9.2.6.2).

```
      SUBROUTINE DVINT(EF, K,VRINT)
      PI=3.1415926535
      RO=SQRT(2./3)
      PI3=PI/3
      FI=-PI3
      VRINT=2.*EF(0.,0.,K)
      DO 10 I=1,6
      FI=FI+PI3
      X=RO*COS(FI)
      Y=RO*SIN(FI)
10    VRINT=VRINT+EF(X,Y,K)
      VRINT=PI/8.*VRINT
      RETURN
      END
```

Main program is of form:

```
C==================================================
C      IZRACUNAVANJE DVOSTRUKOG INTEGRALA
C==================================================
      EXTERNAL EF
      OPEN(6,FILE='DVINT.OUT')
      WRITE (6,5)
5     FORMAT (1H1//10X,'IZRACUNAVANJE DVOSTRUKOG',
     1' INTEGRALA'//)
      DO 10 K=1,3
      CALL DVINT(EF, K, VRINT)
10    WRITE (6,15)K,VRINT
15    FORMAT (15X,I1,' PRIMER'// 10X,
     1 'VREDNOST INTEGRALA =',F12.6//)
      STOP
      END
```

By using this program we calculated approximately values of the following integrals:

$$1^0 \qquad \iint\limits_G \frac{16x^2y^2}{1+x^2+y^2}\, dxdy;$$

$$2^0 \qquad \iint\limits_G \sqrt{1+(1+x)^2+y^2}\, dxdy;$$

$$3^0 \qquad \iint\limits_G \frac{24x^2}{\sqrt{2-x^2-y^2}}\, dxdy.$$

Function subroutine `EF` and output listing are of form:

```
      FUNCTION EF(X,Y,K)
GO TO (10,20,30),K
10    EF=(16.*X*X*Y*Y)/(1.+X*X+Y*Y)
      RETURN
20    EF=SQRT(1.+Y*Y+(1.+X)**2)
      RETURN
30    EF=(24.*X*X)/SQRT(2.-X*X-Y*Y)
      RETURN
      END
```

```
1
          IZRACUNAVANJE DVOSTRUKOG INTEGRALA
                1 PRIMER
          VREDNOST INTEGRALA =    1.256637
                2 PRIMER
          VREDNOST INTEGRALA =    4.858376
                3 PRIMER
          VREDNOST INTEGRALA =   16.324200
```

9.2.7. Packages for Numerical Integration

Numerical integration of both discrete data and known functions are needed in engineering practice. The procedures for first case are based on fitting approximating polynomials to the data and integrating the approximating polynomials. The direct fit polynomial method works well for both equally spaced data and non-equally spaced data. Least squares fit polynomials can be used for large sets of data or sets of rough data. The Newton-Cotes formulas, which are based on Newton forward-difference polynomials, give simple integration formulas for equally spaced data. Romberg integration, which is extrapolation of the trapezoid rule is of important practical use. An example of multiple integration is presented as illustrative case.

Of presented simple methods it is likely that Romberg integration is most efficient. Simpson's rules are elegant, but the first extrapolation of Romberg integration gives comparable results. Subsequent extrapolation of Romberg integration increase the order at a very satisfactory rate. Simpson's rules could be developed into an extrapolation procedure, but with no advantage over Romberg integration.

Many commercial software packages contain solvers for numerical integration. Some of the more prominent systems are `Matlab` and `Mathcad`. More sophisticated systems, such as `Mathematica`, `Macsyma` (VAX UNIX MACSYMA, Reference Manual, Symbolics Inc., Cambridge, MA), and `Maple` (MAPLE V Library Reference Manual, Springer, NY, 1991) also contain numerical integration solvers.

Some organizations have own packages - collection of high-quality routines, like `ACM` (Collected algorithms), `IMSL` (Houston, TX), `NAG` (Numerical Algorithms Group, Downers Grove, IL), and some famous individual packages are `QUADPACK` (R. Piessens, et all.), *QUADPACK, A Subroutine Package for Automatic Integration*, Springer, Berlin, 1983), `CUBTRI` (Cubature Formulae Over Triangle), `SSP` (IBM Numerical Software).

The book *Numerical Recipes* ([4], Chap. 4) contains several subroutines for integration of functions. Some algorithms, from which some are codded, are given in book *Numerical Methods for Engineers and Scientists* ([3], Chap. 6).

On the end, in order to give some hints for software own development or usage of software packages, we will give standard test examples for testing or benchmarking.

Standard test examples (Indefinite integrals):

$$1^0 \int \sin x\,dx;\ 2^0 \int \sqrt{\tan x}\,dx;\ 3^0 \int \frac{x}{x^3-1}\,dx;\ 4^0 \int \frac{x}{\sin^2 x}\,dx;\ 5^0 \int \frac{\log x}{\sqrt{x+1}}\,dx;$$

$$6^0 \int \frac{x}{\sqrt{1+x}+\sqrt{1-x}}\,dx;\ 7^0 \int e^{-ax^2}\,dx;\ 8^0 \int \frac{x}{\log^3 x}\,dx; 9^0 \int \frac{\sin x}{x^2}\,dx;\ 10^0 \int \frac{1}{2+\cos x}\,dx;$$

Standard test examples (Definite integrals):

$$1^0 \int_0^{4\pi} \frac{1}{2+\cos x}\,dx;\ 2^0 \int_{-\infty}^{\infty} \frac{\sin x}{x}\,dx;\ 3^0 \int_0^{\infty} \frac{e^{-x}}{\sqrt{x}}\,dx;\ 4^0 \int_0^{\infty} \frac{x^2 e^{-x}}{1-e^{-2x}}\,dx;\ 5^0 \int_0^{\infty} e^{-x^2}\log^2 x\,dx;$$

$$6^0 \int_1^{\infty} e^{-x}x^3\log^2 x\,dx;\ 7^0 \int_0^{\infty} \frac{x^2}{1+x^3}\,dx;\ 8^0 \int_{-1}^{1} \frac{1}{x^2}\,dx;\ 9^0 \int_1^{\infty} e^{-x}x^{11/3}\,dx;$$

Bibliography (Cited references and further reading)

[1] Milovanović, G.V., *Numerical Analysis II*, Naučna knjiga, Beograd, 1988 (Serbian).

[2] Milovanović, G.V. and Djordjević, Dj.R., *Programiranje numeričkih metoda na FORTRAN jeziku.* Institut za dokumentaciju zaštite na radu "Edvard Kardelj", Niš, 1981 (Serbian).

[3] Hoffman, J.D., *Numerical Methods for Engineers and Scientists.* Taylor & Francis, Boca Raton-London-New York-Singapore, 2001.

[4] Press, W.H., Flannery, B.P., Teukolsky, S.A., and Vetterling, W.T., *Numerical Recepies - The Art of Scientific Computing.* Cambridge University Press, 1989.

[5] Kronrod, A. S., *Nodes and Weights of Quadrature Formulas.* Consultants Bureau, New York, 1965.

[6] Clenshaw, C.W. and Curtis, A.R., *A method for numerical integration on an automatic computer* Num. Math. 2(1960), pp. 197–205.

[7] Engels, H., *Numerical Quadrature and Cubature.* Academic Press, London, 1980.

[8] Abramowitz, M., *On the practical evaluation of integrals.* SIAM J. Appl. Math. 2(1954)20-35.

[9] Davis, P.J., and P. Rabinowitz, P., *Methods of Numerical Integration.* Academic Press, New York, 1975.

[10] Krylov, V.I., *Approximate Calculation of Integrals.* MacMillan, New York, 1962 (Russian, transl. A.H. Stroud).

[11] Stroud, A.H., *Approximative Calculation of Multiple Integrals.* Prentice-Hall, Englewood Cliffs, N.J. 1971.

[12] Mysovskih, I.P., *Interpolyacionnye Kubaturnye Formuly.* Nauka, Moskva, 1981.

[13] Stroud, A.H., *Gaussian Quadrature Formulas.* Prentice Hall, Englewood Cliffs, N.J., 1966.

[14] Ralston,A., *A First Course in Numerical Analysis.*
McGraw-Hill, New York, 1965.

[15] Hildebrand, F.B., *Introduction to Numerical Analysis.*
Mc.Graw-Hill, New York, 1974.

[16] Acton, F.S., *Numerical Methods That Work* (corrected edition). Mathematical Association of America, Washington, D.C., 1990.

[17] Abramowitz, M., and Stegun, I.A., *Handbook of Mathematical Functions.* National Bureau of Standards, Applied Mathematics Series, Washington, 1964 (reprinted 1968 by Dover Publications, New York).

[18] Rice, J.R., *Numerical Methods, Software, and Analysis.* McGraw-Hill, New York, 1983.

[19] Forsythe, G.E., Malcolm, M.A., and Moler, C.B., *Computer Methods for Mathematical Computations.* Englewood Cliffs, Prentice-Hall, NJ, 1977.

[20] Kahaner, D., Moler, C., and Nash, S., 1989, *Numerical Methods and Software.* Englewood Cliffs, Prentice Hall, NJ, 1989.

[21] Hamming, R.W., *Numerical Methods for Engineers and Scientists.* Dover, New York, 1962 (reprinted 1986).

[22] Ferziger, J.H., *Numerical Methods for Engineering Applications.* Stanford University, John Willey & Sons, Inc., New York, 1998.

[23] Pearson, C.E., *Numerical Methods in Engineering and Science.* University of Washington, Van Nostrand Reinhold Company, New York, 1986.

[24] Stephenson, G. and Radmore, P.M., *Advanced Mathematical Methods for Engineering and Science Students.* Imperial College London, University College, London Cambridge Univ. Press, 1999.

[25] Milovanović, G.V. and Kovačević, M.A., *Zbirka rešenih zadataka iz numeričke analize.* Naučna knjiga, Beograd, 1985. (Serbian).

[26] *IMSL Math/Library Users Manual* , IMSL Inc., 2500 City West Boulevard, Houston TX 77042.

[27] *NAG Fortran Library*, Numerical Algorithms Group, 256 Banbury Road, Oxford OX27DE, U.K.

[28] Piessens, R., de Doncker-Kapenga, E., Überhuber, C.W., Kahaner, D.K., *QUADPACK, A Subroutine Package for Automatic Integration.* Springer-Verlag, Berlin, 1983.

[29] Laurie, D. P., *Algorithm 584 CUBTRI: Automatic cubature over a triangle.* ACM Trans. Math. Software 8(1982), 210.–218.

[30] Stoer, J., and Bulirsch, R., *Introduction to Numerical Analysis.* Springer-Verlag, New York, 1980.

[31] Johnson, L.W., and Riess, R.D., *Numerical Analysis, 2nd ed.* Addison- Wesley, Reading, MA, 1982.

[32] Ralston, A., and Rabinowitz, P., *A First Course in Numerical Analysis, 2nd ed.* McGraw-Hill, New York, 1978.

[33] Isaacson, E., and Keller, H.B., *Analysis of Numerical Methods.* Wiley, New York, 1966.